

FILEID**PATFRE

N 8

PPPPPPPP P AAAAAA TTTTTTTT FFFFFFFF RRRRRRRR EEEEEEEE
PPPPPPPP P AAAAAA TTTTTTTT FFFFFFFF RRRRRRRR EEEEEEEE
PP PP AA AA TT FF RR RR EE
PP PP AA AA TT FF RR RR EE
PP PP AA AA TT FF RR RR EE
PPPPPPPP AA AA TT FFFFFF RRRRRRRR EEEEEEEE
PPPPPPPP AA AA TT FFFFFF RRRRRRRR EEEEEEEE
PP AAAAAAAA TT FF RR RR EE
PP AAAAAAAA TT FF RR RR EE
PP AA AA TT FF RR RR EE
PP AA AA TT FF RR RR EE
PP AA AA TT FF RR RR EEEEEEEE
PP AA AA TT FF RR RR EEEEEEEE

....
....
....
....

LL IIIII SSSSSSS
LL IIIII SSSSSSS
LL II SS SSS
LLLLLLLLL LIII SSSSSSS
LLLLLLLLL LIII SSSSSSS

PA
VO

```
1 0001 0 MODULE PATFRE (
2 0002 0           %IF %VARIANT EQL 1
3 0003 0           %THEN
4 0004 0               ADDRESSING_MODE (EXTERNAL = LONG_RELATIVE, NONEXTERNAL = LONG_RELATIVE),
5 0005 0           %FI
6 0006 0           IDENT = 'V04-000') =
7 0007 1 BEGIN
8 0008 1
9 0009 1 !*****+
10 0010 1 *
11 0011 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
12 0012 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
13 0013 1 * ALL RIGHTS RESERVED.
14 0014 1 *
15 0015 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
16 0016 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
17 0017 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
18 0018 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
19 0019 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
20 0020 1 * TRANSFERRED.
21 0021 1 *
22 0022 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
23 0023 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
24 0024 1 * CORPORATION.
25 0025 1 *
26 0026 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
27 0027 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
28 0028 1 *
29 0029 1 *
30 0030 1 !*****+
31 0031 1
32 0032 1 FACILITY: PATCH
33 0033 1
34 0034 1 ++
35 0035 1 FUNCTIONAL DESCRIPTION:
36 0036 1
37 0037 1     Free storage allocator and manager for symbol table.
38 0038 1
39 0039 1 Version: V02-008
40 0040 1
41 0041 1 History:
42 0042 1     Author:
43 0043 1         Isaac Nassi, 7 Jul 1976: Version 01
44 0044 1
45 0045 1 MODIFIED BY:
46 0046 1
47 0047 1     V02-008 PCG0001      Peter George    02-FEB-1981
48 0048 1         Add require statement for LIB$:PATDEF.REQ
49 0049 1
50 0050 1     Modified by:
51 0051 1         Carol Peters, 11 Oct 1977: Version 13
52 0052 1
53 0053 1     Modified by:
54 0054 1         Kathleen Morse, 13 Oct 1977: Version 14
55 0055 1
56 0056 1     Revision history:
57 0057 1
```

	NO	DATE	PROGRAMMER	PURPOSE
58	0058	1	NO	-----
59	0059	1	--	-----
60	0060	1		
61	0061	1	00	13-OCT-77 K.D. MORSE MODIFY VERSION 13 FOR PATCH
62	0062	1	01	27-DEC-77 K.D. MORSE ADD ROUTINE PAT\$REPORT FREE.
63	0063	1	02	27-DEC-77 K.D. MORSE INITIALIZE PAT\$GL RST BEGN.
64	0064	1	03	5-JAN-78 K.D. MORSE NO CHANGES FOR VERS 14-16.
65	0065	1	04	14-APR-78 K.D. MORSE NO CHANGES FOR VERS 17.
66	0066	1	05	25-APR-78 K.D. MORSE CONVERT TO NATIVE COMPILER.
67	0067	1	06	18-MAY-78 K.D. MORSE NO CHANGES FOR VERS 18-19.
68	0068	1	07	13-JUN-78 K.D. MORSE ADD FAO COUNTS TO SIGNALS.
69	0069	1		
70	0070	1	--	

; R

```

72 0071 1 ++
73 0072 1 Abstract:
74 0073 1
75 0074 1 THIS MODULE CONTAINS PROCEDURES TO MANAGE AN AREA OF FREE
76 0075 1 STORAGE.
77 0076 1
78 0077 1 THE ROUTINE PAT$FREEINIT IS CALLED AT SYSTEM INITIALIZATION
79 0078 1 TO INITIALIZE THE MODULE. IT TAKES AS INPUT THE ADDRESS
80 0079 1 OF THE PROCEDURE TO BE CALLED IN THE EVENT AN ERROR
81 0080 1 IS ENCOUNTERED. CODES FOR ERROR CONDITIONS ARE CONTAINED IN
82 0081 1 FILE 'PATMSG.REQ'.
83 0082 1
84 0083 1 THE ROUTINE FREEGET IS CALLED TO ALLOCATE A USER SPECIFIED
85 0084 1 NUMBER OF LONGWORDS. IF THE REQUEST CAN BE SATISFIED,
86 0085 1 FREEGET RETURNS A POINTER TO THE BLOCK OF LONGWORDS.
87 0086 1 THE RETURNED BLOCK CONTAINS ONE EXTRA WORD CONTAINING
88 0087 1 THE SIZE REQUEST AT WORD -1 IN THE BLOCK. THIS SIZE
89 0088 1 IS CHECKED WHEN THE STORAGE IS RETURNED. IT SHOULD
90 0089 1 NOT BE MODIFIED. THE ALGORITHM USED IS A FIRST FIT
91 0090 1 ALGORITHM WHICH WHILE NOT OPTIMAL SHOULD GIVE REASONABLE
92 0091 1 RESULTS, AND DO A MINIMUM OF SEARCHING.
93 0092 1
94 0093 1 Dynamic storage is the last 64K bytes of the per process
95 0094 1 address space. It is made accessible by a $CRETVA system
96 0095 1 service call made in FREEINIT.
97 0096 1
98 0097 1 PAT$FREERELEASE IS CALLED TO RELEASE STORAGE NO LONGER NEEDED.
99 0098 1 IT ATTEMPTS TO DO AS MUCH COMPACTION AS IS POSSIBLE.
100 0099 1
101 0100 1 ROUTINE PAT$FREEZ IS CALLED TO ALLOCATE A BLOCK OF
102 0101 1 CLEARED FREE STORAGE.
103 0102 1
104 0103 1 ROUTINE PAT$REPORT_FREE REPORTS THE NUMBER OF BYTES LEFT IN FREE STORAGE.
105 0104 1
106 0105 1 --
107 0106 1
108 0107 1
109 0108 1 TABLE OF CONTENTS
110 0109 1
111 0110 1
112 0111 1 FORWARD ROUTINE
113 0112 1 PAT$FREEINIT : NOVALUE,
114 0113 1 FREEGET,
115 0114 1 PAT$FREERELEASE : NOVALUE,
116 0115 1 PAT$FREEZ,
117 0116 1 PAT$REPORT_FREE;
118 0117 1
119 0118 1
120 0119 1 INCLUDE FILES
121 0120 1
122 0121 1
123 0122 1 LIBRARY 'SYSSLIBRARY:STARLET.L32';
124 0123 1 REQUIRE 'SRC$:PATPCT.REQ';
125 0124 1 REQUIRE 'SRC$:VXSMAC.REQ';
126 0125 1 REQUIRE 'SRC$:PATGEN.REQ';
127 0126 1 REQUIRE 'SRC$:BSTRU.C.REQ';
128 0127 1 REQUIRE 'LIBS:PATDEF.REQ';
129 0128 1
130 0129 1 ! Defines Literals

```

PATFRE
V04-000

E 9
16-Sep-1984 00:16:31 VAX-11 BLISS-32 V4.0-742
14-Sep-1984 12:52:32 DISK\$VMSMASTER:[PATCH.SRC]PATFRE.B32;1 Page 4 (2)

: 129 0580 1 REQUIRE 'LIB\$:PATMSG.REQ';
: 130 0754 1 REQUIRE 'SRC\$:SYSSER.REQ';

PAT
V04

: R

PATFRE
V04-000

F 9
16-Sep-1984 00:16:31 VAX-11 Bliss-32 V4.0-742 Page 5
15-Sep-1984 22:50:49 -\$255\$DUA28:[PATCH.SRC]SYSSE.REQ;1 (1)

```
R0786 1      SWITCHES LIST (SOURCE);  
R0787 1  
R0788 1      EXTERNAL ROUTINE  
R0789 1          PAT$fao_out;           ! formats a line and outputs to the terminal  
R0790 1
```

```
: 131      0836 1
: 132      0837 1 | EXTERNAL REFERENCES
: 133      0838 1
: 134      0839 1
: 135      0840 1
: 136      0841 1 | EXTERNAL
: 137      0842 1     PAT$GL_RST_BEGN,
: 138      0843 1     PAT$GL_ERRCODE;
: 139      0844 1
: 140      0845 1 | EQUATED SYMBOLS
: 141      0846 1
: 142      0847 1
: 143      0848 1 | LITERAL
: 144      0849 1     DEBUGGER = TRUE;          ! COMPILE TIME CONDITIONAL FOR DEBUGGING
: 145
: 146      L 0851 1 %IF DEBUGGER
: 147      %THEN
: 148      0852 1
: 149      0853 1 | LITERAL
: 150      0854 1     NUM_OF_WRK_PAG=128,
: 151      0855 1     STORESIZE=7512*NUM_OF_WRK_PAG)/4;    ! NUMBER OF WORK AREA PAGES
: 152      U 0856 1 %ELSE
: 153      U 0857 1 | LITERAL
: 154      U 0858 1     SYM_TAB_START    =%X'7FFF0000',
: 155      U 0859 1     SYM_TAB_END     =%X'7FFFFFFF';
: 156      U 0860 1     STORESIZE       = (SYM_TAB_END - SYM_TAB_START) / 4;    ! GET 65K OF WORK AREA FROM P0 NOT P1
: 157      0861 1 %FI
: 158
: 159      0862 1
: 160      0863 1 | OWN STORAGE
: 161      0864 1
: 162      0865 1
: 163      L 0866 1 OWN
: 164      L 0867 1 %IF DEBUGGER
: 165      0868 1 %THEN
: 166      0869 1 | PAT$GL_STORAGE: VECTOR[STORESIZE],          ! SET ASIDE WORK AREA IN P0
: 167      0870 1
: 168      0871 1 %FI
: 169      0872 1 | PAT$STORE: REF VECTOR,
: 170      0873 1     PAT$FREELIST : VECTOR [2];          ! pointer to free storage area
: 171      P 0874 1
: 172      P 0875 1 BASED_STRUCTURE (FR. 2.
: 173      P 0876 1     SIZE: 0: 0: 32: 0;
: 174      P 0877 1     NEXT: 1: 0: 32: 0;
```

```

174 0878 1 GLOBAL ROUTINE PAT$FREEINIT : NOVALUE =
175 0879 1
176 0880 1 ++
177 0881 1 FUNCTIONAL DESCRIPTION:
178 0882 1
179 0883 1 PAT$FREEINIT IS CALLED TO INITIALIZE THE FREE STORAGE
180 0884 1 PACKAGE.
181 0885 1
182 0886 1 CALLING SEQUENCE:
183 0887 1
184 0888 1 PAT$FREEINIT ()
185 0889 1
186 0890 1 INPUTS:
187 0891 1
188 0892 1     none
189 0893 1
190 0894 1 IMPLICIT INPUTS:
191 0895 1
192 0896 1     NONE
193 0897 1
194 0898 1 OUTPUTS:
195 0899 1
196 0900 1     NONE
197 0901 1
198 0902 1 IMPLICIT OUTPUTS:
199 0903 1
200 0904 1     PAT$FREELIST AND THE FIRST ELEMENT OF PAT$STORE ARE INITIALIZED.
201 0905 1
202 0906 1 ROUTINE VALUE:
203 0907 1
204 0908 1     NOVALUE
205 0909 1
206 0910 1 SIDE EFFECTS:
207 0911 1
208 0912 1     NONE
209 0913 1
210 0914 1
211 0915 2 BEGIN
212 0916 2
213 0917 2 ++
214 0918 2 THIS ROUTINE HAS TWO VERSIONS, DEPENDING UPON WHETHER IT MUST LEAVE THE
215 0919 2 P1 SPACE FOR THE DEBUGGER TO USE, OR CAN USE IT FOR ITS OWN PURPOSES.
216 0920 2 IF THE DEBUGGER IS TO BE LINKED WITH PATCH, THEN THE COMPILE-TIME VARIABLE "DEBUGGER"
217 0921 2 IS SET TO TRUE, CAUSING PATCH TO ALLOCATE THE WORK SPACE FROM P0 SPACE
218 0922 2 INSTEAD OF P1 SPACE. OTHERWISE, "DEBUGGER" IS SET TO FALSE AND PATCH
219 0923 2 USES THE P1 SPACE FOR WORK AREA.
220 0924 2
221 0925 2
222 0926 2 ++
223 0927 2 FIRST HANDLE THE CASE WHERE THE DEBUGGER IS NOT TO BE LINKED IN.
224 0928 2
225 L 0929 2 ++
226 U 0930 2 %IF NOT DEBUGGER
227 U 0931 2 %THEN
228 U 0932 2 LITERAL
229 U 0933 2     START_TAB      = 0,
230 U 0934 2             END_TAB       = 1;

```

```
: 231 U 0935 2 LOCAL      SYMTAB_DESC : VECTOR [2];           ! descriptor for space to create
: 232 U 0936 2
: 233 U 0937 2
: 234 U 0938 2
: 235 U 0939 2 SYMTAB_DESC [START_TAB] = SYM_TAB_START;
: 236 U 0940 2 SYMTAB_DESC [END_TAB] = SYM_TAB_END;
: 237 U 0941 2 PAT$GL_ERRCODE = $CRETVA (INADR = SYMTAB_DESC, RETADR = SYMTAB_DESC);
: 238 U 0942 2 IF NOT .PAT$GL_ERRCODE
: 239 U 0943 2 THEN
: 240 U 0944 2     SIGNAL (.PAT$GL_ERRCODE);
: 241 U 0945 2
: 242 U 0946 2
: 243 U 0947 2     The create virtual page system service was successful.
: 244 U 0948 2     Now initialize the first few words of the space, and the
: 245 U 0949 2     freelist values.
: 246 U 0950 2
: 247 U 0951 2     PAT$STORE = .SYMTAB_DESC [START_TAB];
: 248 U 0952 2
: 249 U 0953 2
: 250 U 0954 2     NOW HANDLE THE CASE WHERE THE DEBUGGER IS LINKED IN.
: 251 U 0955 2
: 252 U 0956 2
: 253 U 0957 2     %ELSE
: 254 U 0958 2     PAT$STORE = PAT$GL_STORAGE;
: 255 U 0959 2     %FI
: 256 U 0960 2
: 257 U 0961 2     FR_SIZE (.PAT$STORE) = STORESIZE;
: 258 U 0962 2     FR_NEXT (.PAT$STORE) = NULL;
: 259 U 0963 2     FR_NEXT (PAT$FREELIST) = .PAT$STORE;
: 260 U 0964 2     FR_SIZE (PAT$FREELIST) = 2;
: 261 U 0965 2
: 262 U 0966 2
: 263 U 0967 2     NOW SET THE ADDRESS OF THE START OF THE RST.
: 264 U 0968 2
: 265 U 0969 2     PAT$GL_RST_BEGN = .PAT$STORE;
: 266 U 0970 2
: 267 U 0971 1 END;
```

```
.TITLE PATFRE
.IDENT \V04-000\

.PSECT _PAT$OWN,NOEXE,2

00000 PAT$GL_STORAGE:
.BLKB 65536

10000 PAT$STORE:
.BLKB 4

10004 PAT$FREELIST:
.BLKB 8

.EXTRN PAT$FAO_OUT, PAT$GL_RST_BEGN
.EXTRN PAT$GL_ERRCODE
.WEAK ACCESS_CHECK

.PSECT _PAT$CODE,NOWRT,2
```

52 00000000:	EF 9E 00002	0004 00000	.ENTRY PAT\$FREEINIT, Save R2	: 0878
62 00000000:	EF 9E 00009	62 D 00010	MOVAB PAT\$STORE, R2	: 0958
50		8F 3C 00013	MOVAB PAT\$GL_STORAGE, PAT\$STORE	: 0961
60 4000	04	A0 D4 00018	MOVL PAT\$STORE, R0	: 0962
08 A2		50 D0 0001B	MOVZWL #16384, (R0)	: 0963
04 A2		02 D0 0001F	CLRL 4(R0)	: 0964
00000000G EF		50 D0 00023	MOVL R0, PAT\$FREELIST+4	: 0969
		04 0002A	MOVL R0, #2, PAT\$FREELIST	: 0971
			MOVL R0, PAT\$GL_RST_BEGN	
			RET	

; Routine Size: 43 bytes, Routine Base: _PAT\$CODE + 0000

: 269 0972 1 ROUTINE FREEGET (NEED1) =
270 0973 1
271 0974 1 ++
272 0975 1 FUNCTIONAL DESCRIPTION:
273 0976 1
274 0977 1 PROCEDURE FREEGET IS CALLED WITH ONE INPUT ARGUMENT,
275 0978 1 NEED1, WHICH REPRESENTS THE NUMBER OF WORDS OF FREE
276 0979 1 STORAGE NEEDED. IT SEARCHES THE FREELIST FOR A
277 0980 1 NODE THAT HAS AT LEAST THE REQUIRED NUMBER OF WORDS.
278 0981 1 IF THE NODE HAS MORE THAN THE REQUIRED NUMBER OF WORDS,
279 0982 1 THE NODE IS SPLIT INTO TWO NODES, AND ONE IS RETURNED.
280 0983 1
281 0984 1 CALLING SEQUENCE:
282 0985 1
283 0986 1 FREEGET ()
284 0987 1
285 0988 1 INPUTS:
286 0989 1
287 0990 1 NEED1 - THE NUMBER OF WORDS NEEDED OF FREE STORAGE
288 0991 1
289 0992 1 IMPLICIT INPUTS:
290 0993 1
291 0994 1 THE CURRENT STATE OF PAT\$FREELIST.
292 0995 1
293 0996 1 OUTPUTS:
294 0997 1
295 0998 1 THE ADDRESS OF THE FREE STORAGE ACQUIRED.
296 0999 1
297 1000 1 IMPLICIT OUTPUTS:
298 1001 1
299 1002 1 THE STATE OF PAT\$FREELIST AND THE COUNT OF WORDS
300 1003 1 OF STORAGE LEFT ARE CHANGED.
301 1004 1
302 1005 1 ROUTINE VALUE:
303 1006 1
304 1007 1 THE ADDRESS OF THE BLOCK
305 1008 1
306 1009 1 SIDE EFFECTS:
307 1010 1
308 1011 1 IF THERE IS NO FREE STORAGE, THEN A SEVERE ERROR IS REPORTED WHICH
309 1012 1 CAUSES AN "UNWIND" FOR THE NEXT COMMAND (INTERACTIVE MODE) OR AN
310 1013 1 EXIT FROM PATCH (BATCH MODE).
311 1014 1 --
312 1015 1
313 1016 2 BEGIN
314 1017 2
315 1018 2 LOCAL
316 1019 2 NEED,
317 1020 2 OLDOMNode,
318 1021 2 LIST,
319 1022 2 HAVE;
320 1023 2
321 1024 2 NEED = .NEED1 + 1 ; ! ONE WORD BIAS FOR SIZE
322 1025 2 IF .NEED LSS 0
323 1026 2 THEN
324 1027 2 SIGNAL (PAT\$ NOFREE);
325 1028 2 LIST = .FR_NEXT (PAT\$FREELIST);

```

: 326 1029 2 OLDDNODE = PAT$FREELIST;
: 327 1030 2 WHILE .LIST NEQ NULL
: 328 1031 2 DO
: 329 1032 3 BEGIN
: 330 1033 4 IF ((HAVE = .FR_SIZE (.LIST)) EQ .NEED) ! FIND FIRST FIT
: 331 1034 4 OR (.have GEQ (.need + 2))
: 332 1035 3 THEN
: 333 1036 4 BEGIN
: 334 1037 4 IF .HAVE EQ .NEED ! FOUND
: 335 1038 4 THEN FR_NEXT (.OLDDNODE) = .FR_NEXT (.LIST)
: 336 1039 4 ELSE
: 337 1040 5 BEGIN ! LARGER THAN NEEDED
: 338 1041 5 FR_NEXT (.LIST + .NEED * 4) = .FR_NEXT (.LIST);
: 339 1042 5 FR_SIZE (.LIST + .NEED * 4) = .FR_SIZE (.LIST) - .NEED;
: 340 1043 5 FR_NEXT (.OLDDNODE) = .LIST + .NEED * 4;
: 341 1044 4 END; ! LARGER THAN NEEDED
: 342 1045 4 FR_SIZE (.LIST) = .NEED;
: 343 1046 4
: 344 1047 4 RETURN (.LIST + 4); ! CONTAINS SIZE
: 345 1048 4 END ! FOUND
: 346 1049 3 ELSE LIST = .FR_NEXT (.LIST);
: 347 1050 3 OLDDNODE = .FR_NEXT (.OLDDNODE);
: 348 1051 2 END; ! FIND FIRST FIT
: 349 1052 2 SIGNAL (PAT$_NOFREE); ! Signal error
: 350 1053 2 RETURN NULL
: 351 1054 1 END;

```

			003C 00000 FREEGET: .WORD				
52	04	AC 00000000G	00 9E 00002	MOVAB	Save R2,R3,R4,R5		0972
			01 C1 00009	ADDL3	#1, NEED1, NEED		1024
			09 18 0000E	BGEQ	1\$		1025
		006D8112	8F DD 00010	PUSHL	#7176466		1027
			01 FB 00016	CALLS	#1, LIB\$SIGNAL		
		51 00000000'	EF D0 00019	1\$:	MOVL PAT\$FREELIST+4, LIST		1028
		53 00000000'	EF 9E 00020	MOVAB	PAT\$FREELIST, OLDDNODE		1029
			51 D5 00027	2\$:	TSTL LIST		1030
			40 13 00029	BEQL	7\$		
		54	61 D0 0002B	MOVL	(LIST), HAVE		1033
		52	54 D1 0002E	CMPL	HAVE, NEED		
			09 13 00031	BEQL	3\$		
	50	02	A2 9E 00033	MOVAB	2(R2), R0		1034
	50		54 D1 00037	CMPL	HAVE, R0		
			25 19 0003A	BLSS	6\$		
		52	54 D1 0003C	3\$:	CMPL	HAVE, NEED	1037
			07 12 0003F	BNEQ	4\$		
	04	A3	04 A1 D0 00041	MOVL	4(LIST), 4(OLDDNODE)		1038
			11 11 00046	BRB	5\$		
60	04	50 6142 DE 00048	4\$:	MOVAL	(LIST)[NEED], R0		1041
	04	A0	04 A1 D0 0004C	MOVL	4(LIST), 4(R0)		
		61	52 C3 00051	SUBL3	NEED, (LIST), (R0)		1042
	04	A3	50 D0 00055	MOVL	R0, 4(OLDDNODE)		1043
		61	52 D0 00059	5\$:	NEED, (LIST)		1045
		50	04 A1 9E 0005C	MOVAB	4(LIST), R0		1047

51	04	A1	04 00060	RET				
53	04	A3	00 00061	6\$: MOVL	4(LIST), LIST			1049
			00 00065	MOVL	4(OLDNODE), OLDNODE			1050
		BC	11 00069	BRB	2\$			1030
65	006D8112	8F	DD 0006B	7\$: PUSHL	#7176466			1052
		01	FB 00071	CALLS	#1, LIB\$SIGNAL			1053
		50	D4 00074	CLRL	R0			1054
		04	00076	RET				

; Routine Size: 119 bytes, Routine Base: _PAT\$CODE + 002B

: 353 1055 1 GLOBAL ROUTINE PAT\$FREERELease (INODE1, NUM1) : NOVALUE =
354 1056 1
355 1057 1 ++
356 1058 1 FUNCTIONAL DESCRIPTION:
357 1059 1
358 1060 1 PROCEDURE PAT\$FREERELease IS CALLED TO RETURN STORAGE
359 1061 1 BACK INTO THE FREE STORAGE POOL. IT MAINTAINS A
360 1062 1 FREELIST IN ASCENDING ORDER OF STORAGE ADDRESSES,
361 1063 1 AND FINDS THE PROPER LOCATION FOR INODE IN THIS
362 1064 1 LIST. THEN IT ATTEMPTS TO MERGE WITH THE LEFT HAND
363 1065 1 NEIGHBOR AND WITH THE RIGHT HAND NEIGHBOR.
364 1066 1
365 1067 1 CALLING SEQUENCE:
366 1068 1
367 1069 1 PAT\$FREERELease ()
368 1070 1
369 1071 1 INPUTS:
370 1072 1
371 1073 1 INODE1 - ADDRESS OF A NODE
372 1074 1 NUM1 - LENGTH OF A NODE
373 1075 1
374 1076 1 IMPLICIT INPUTS:
375 1077 1
376 1078 1 THE FREE STORAGE POOL, AND THE CURRENT CONTENTS OF PAT\$FREELIST.
377 1079 1
378 1080 1 OUTPUTS:
379 1081 1
380 1082 1 NONE
381 1083 1
382 1084 1 IMPLICIT OUTPUTS:
383 1085 1
384 1086 1 AN ERROR MESSAGE CALL ON ERROR
385 1087 1
386 1088 1 ROUTINE VALUE:
387 1089 1
388 1090 1 NOVALUE
389 1091 1
390 1092 1 SIDE EFFECTS:
391 1093 1
392 1094 1 THE STORAGE IS RETURNED TO THE POOL.
393 1095 1 --
394 1096 1
395 1097 2 BEGIN
396 1098 2
397 1099 2 LOCAL
398 1100 2 INODE,
399 1101 2 NUM,
400 1102 2 NODE,
401 1103 2 OLDDNODE;
402 1104 2
403 1105 2 INODE = .INODE1 - 4 ; ! BIAS FOR SIZE WORD
404 1106 2 NUM = .NUM1 + 1 ; ! INVISIBLE TO USER
405 1107 2 IF .INODE GEQA .PAT\$STORE
406 1108 2 AND .INODE LEQA .PAT\$STORE + (STORESIZE * 4) - 1 ! CORRECT SIZE
407 1109 2 AND .FR_SIZE (.INODE) EQL .NUM
408 1110 2 THEN
409 1111 3 BEGIN ! IN RANGE

```

410      1112 3      NODE = .FR_NEXT (PAT$FREELIST);
411      1113 3      OLDDNODE = PAT$FREELIST;
412      1114 3      WHILE .INODE GTRA .NODE AND .NODE NEQ NULL
413      1115 3      DO
414          1116 4      BEGIN
415          1117 4      OLDDNODE = .NODE;
416          1118 4      NODE = .FR_NEXT (.NODE);
417          1119 3      END;
418          1120 3      FR_NEXT (.INODE) = .NODE;
419          1121 3      FR_NEXT (.OLDDNODE) = .INODE;
420          1122 3      INCR I FROM 1 TO 2 DO
421          1123 3      IF .OLDDNODE + .FR_SIZE (.OLDDNODE) * 4 EQLA .FR_NEXT (.OLDDNODE)
422          1124 3      AND .OLDDNODE NEQA PAT$FREELIST !NOT FIRST ON LIST
423          1125 3      THEN
424              1126 4      BEGIN
425              1127 4      FR_SIZE (.OLDDNODE) = .FR_SIZE (.OLDDNODE) + .FR_SIZE (.FR_NEXT (.OLDDNODE));
426              1128 4      FR_NEXT (.OLDDNODE) = .FR_NEXT (.FR_NEXT (.OLDDNODE));
427              1129 4      END
428          1130 3      ELSE OLDDNODE = .FR_NEXT (.OLDDNODE);
429          1131 3      END
430          1132 2      ELSE
431              1133 3      BEGIN
432              1134 3      SIGNAL (IF .FR_SIZE (.INODE) NEQ .NUM
433                  1135 3      THEN PATS_FREESIZE
434                  1136 3      ELSE PATS_FREERANGE); ! alarm
435          1137 2      END;
436      1138 1      END;

```

					.ENTRY	PAT\$FREERELEASE, Save R2,R3	1055
52	04	53 00000000'	EF 9E 00002		MOVAB	PAT\$STORE R3	1105
51	08	AC	04 C3 00009		SUBL3	#4, INODE1, INODE	1106
		AC	01 C1 0000E		ADDL3	#1, NUM1, NUM	1107
		63	52 D1 00013		CMPL	INODE, PAT\$STORE	
			65 1F 00016		BLSSU	6\$	
50		63 0000FFFF	8F C1 00018		ADDL3	#65535, PAT\$STORE, R0	1108
		50	52 D1 00020		CMPL	INODE, R0	
			58 1A 00023		BGTRU	6\$	
		51	62 D1 00025		CMPL	(INODE), NUM	1109
			53 12 00028		BNEQ	6\$	
		50	08 A3 D0 0002A		MOVL	PAT\$FREELIST+4, NODE	1112
		51	04 A3 9E 0002E		MOVAB	PAT\$FREELIST, OLDDNODE	1113
		50	52 D1 00032	1\$:	CMPL	INODE, NODE	1114
			0D 1B 00035		BLEQU	2\$	
			50 D5 00037		TSTL	NODE	
			09 13 00039		BEQL	2\$	
		51	50 D0 0003B		MOVL	NODE, OLDDNODE	1117
		50	04 A0 D0 0003E		MOVL	4(NODE), NODE	1118
			EE 11 00042		BRB	1\$	1114
04	A2		50 D0 00044	2\$:	MOVL	NODE, 4(INODE)	1120
04	A1		52 D0 00048		MOVL	INODE, 4(OLDDNODE)	1121
52			01 D0 0004C		MOVL	#1, I	1122
50			61 D0 0004F	3\$:	MOVL	(OLDDNODE), R0	1123
50		6140	DE 00052		MOVAL	(OLDDNODE)[R0], R0	

	04	A1	50	D1	00056	CMPL	R0, 4(OLDNODE)	:
	50	04	A3	9E	0005A	BNEQ	4\$	1124
	50		51	D1	00060	MOVAB	PAT\$FREELIST, R0	
			0F	13	00063	CMPL	OLDNODE, R0	
			61	04	B1	BEQL	4\$	1127
			50	04	A1	ADDL2	04(OLDNODE), (OLDNODE)	
	04	A1	04	A0	D0	MOVL	4(OLDNODE), R0	1128
				04	0006D	MOVL	4(R0), 4(OLDNODE)	
D3				04	11	BRB	5\$	1123
			51	04	A1	MOVL	4(OLDNODE), OLDNODE	
			52	02	F3	AOBLEQ	#2, I, 3\$	1130
				04	0007C	RET		1123
			51	62	D1	CMPL	(INODE), NUM	
				08	0007D	BEQL	7\$	1107
			006D80BA	8F	DD	PUSHL	#7176378	
				06	00080	BRB	8\$	
			006D80B2	8F	DD	PUSHL	#7176370	
	00000000G	00		01	0008A	CALLS	#1, LIB\$SIGNAL	
				04	00090	RET		1138
					8\$:			

: Routine Size: 152 bytes, Routine Base: _PAT\$CODE + 00A2

```

438      1139 1 GLOBAL ROUTINE PAT$FREEZ (NEED) =
439      1140 1
440      1141 1 +++
441      1142 1 FUNCTIONAL DESCRIPTION:
442      1143 1
443      1144 1 CALLS FREEGET TO ALLOCATE STORAGE, AND CLEARS IT
444      1145 1
445      1146 1 CALLING SEQUENCE:
446      1147 1
447      1148 1 PAT$FREEZ ()
448      1149 1
449      1150 1 INPUTS:
450      1151 1
451      1152 1 LENGTH OF AREA WANTED (IN LONGWORDS)
452      1153 1
453      1154 1 IMPLICIT INPUTS:
454      1155 1
455      1156 1 NONE
456      1157 1
457      1158 1 OUTPUTS:
458      1159 1
459      1160 1 ADDRESS OF AREA
460      1161 1
461      1162 1 IMPLICIT OUTPUTS:
462      1163 1
463      1164 1 NONE
464      1165 1
465      1166 1 ROUTINE VALUE:
466      1167 1
467      1168 1 NOVALUE
468      1169 1
469      1170 1 SIDE EFFECTS:
470      1171 1
471      1172 1 NONE
472      1173 1 --
473      1174 1
474      1175 2 BEGIN
475      1176 2
476      1177 2 LOCAL
477      1178 2 P;
478      1179 2
479      1180 2 IF (P = FREEGET (.NEED)) NEQ 0
480      1181 2 THEN ZEROCOR (.P, .NEED);
481      1182 2 RETURN .P
482      1183 1 END;

```

				04	007C 00000	.ENTRY	PAT\$FREEZ, Save R2,R3,R4,R5,R6	1139
		FEE7	CF		01 DD 00002	PUSHL	NEED	1180
			56		01 FB 00005	CALLS	#1, FREEGET	
					50 D0 0000A	MOVL	R0, P	
					0B 15 0000D	BEQL	1\$	
					02 78 0000F	ASHL	#2, NEED, R0	
50	50	04	AC		00 2C 00014	MOVCS	#0, (SP), #0, R0, (P)	1181
			6E					

PATFRE
V04-000

E 10
16-Sep-1984 00:16:31 VAX-11 Bliss-32 v4.0-742
14-Sep-1984 12:52:32 DISK\$VMSMASTER:[PATCH.SRC]PATFRE.B32;1 Page 17
(6)

50 66 00019
56 D0 0001A 1\$: MOVL P, R0
04 0001D RET

; 1182
; 1183

: Routine Size: 30 bytes, Routine Base: _PAT\$CODE + 013A

```
484 1184 1 GLOBAL ROUTINE PAT$REPORT_FREE =  
485 1185 1  
486 1186 1 ++  
487 1187 1 FUNCTIONAL DESCRIPTION:  
488 1188 1  
489 1189 1 REPORTS THE NUMBER OF BYTES LEFT IN FREE STORAGE.  
490 1190 1  
491 1191 1 CALLING SEQUENCE:  
492 1192 1  
493 1193 1 PAT$REPORT_FREE ()  
494 1194 1  
495 1195 1 INPUTS:  
496 1196 1  
497 1197 1 NONE  
498 1198 1  
499 1199 1 IMPLICIT INPUTS:  
500 1200 1  
501 1201 1 THE ELEMENTS OF THE FREE LIST.  
502 1202 1  
503 1203 1 OUTPUTS:  
504 1204 1  
505 1205 1 NONE  
506 1206 1  
507 1207 1 IMPLICIT OUTPUTS:  
508 1208 1  
509 1209 1 NONE  
510 1210 1  
511 1211 1 ROUTINE VALUE:  
512 1212 1  
513 1213 1 THE NUMBER OF BYTES OF STROAGE THAT IS FREE.  
514 1214 1  
515 1215 1 SIDE EFFECTS:  
516 1216 1  
517 1217 1 NONE  
518 1218 1 --  
519 1219 1  
520 1220 2 BEGIN  
521 1221 2  
522 1222 2 LOCAL  
523 1223 2 COUNT,  
524 1224 2 POINTER;  
525 1225 2  
526 1226 2 COUNT = 0;  
527 1227 2  
528 1228 2 ++  
529 1229 2 STEP THROUGH THE FREE LIST.  
530 1230 2!--  
531 1231 2 POINTER = .FR_NEXT (PAT$FREELIST);  
532 1232 2 WHILE .POINTER NEQ NULL  
533 1233 2 DO  
534 1234 3 BEGIN  
535 1235 3 COUNT = .COUNT + .FR_SIZE (.POINTER);  
536 1236 3 POINTER = .FR_NEXT (.POINTER);  
537 1237 2 END;  
538 1238 2 RETURN .COUNT * 4  
539 1239 1 END;
```

		0000 00000	.ENTRY	PAT\$REPORT_FREE, Save nothing	: 1184
51	00000000'	50 D4 00002	CLRL	COUNT	: 1226
		EF D0 00004	MOVL	PAT\$FREELIST+4, POINTER	: 1231
		09 13 0000B 1\$:	BEQL	2\$: 1232
50		61 C0 0000D	ADDL2	(POINTER), COUNT	: 1235
51	04	A1 D0 00010	MOVL	4(POINTER), POINTER	: 1236
50		F5 11 00014	BRB	1\$: 1232
		04 C4 00016 2\$:	MULL2	#4, R0	: 1238
		04 00019	RET		: 1239

; Routine Size: 26 bytes, Routine Base: _PAT\$CODE + 0158

PATFRE
V04-000

H 10
16-Sep-1984 00:16:31 VAX-11 Bliss-32 v4.0-742
14-Sep-1984 12:52:32 DISK\$VMSMASTER:[PATCH.SRC]PATFRE.B32;
Page 20
(8)

: 541 1240 1 END
: 542 1241 0 ELUDOM

.EXTRN LIB\$SIGNAL

PSECT SUMMARY

Name	Bytes	Attributes
PAT\$DOWN	65548 NOVEC, WRT, RD ,NOEXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)	
PAT\$CODE	370 NOVEC,NOWRT, RD , EXE,NOSHR, LCL, REL, CON,NOPIC,ALIGN(2)	

Library Statistics

File	----- Symbols -----			Pages Mapped	Processing Time
	Total	Loaded	Percent		
\$_255\$DUA28:[SYSLIB]STARLET.L32;1	9776	4	0	581	00:01.0

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/VARIANT:1/LIS=LISS:PATFRE/OBJ=OBJ\$:PATFRE MSRC\$:PATFRE/UPDATE=(ENH\$:PATFRE)

Size: 370 code + 65548 data bytes
Run Time: 00:18.7
Elapsed Time: 00:53.2
Lines/CPU Min: 3979
Lexemes/CPU-Min: 48086
Memory Used: 117 pages
Compilation Complete

0301 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

PATERR
LIS

PATEXA
LIS

PATFRE
LIS

PATIHD
LIS

PATINS
LIS

PATINT
LIS